BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1A shows paths of execution;
FIG. 1B shows the comparative costs of compiling dominant paths;
FIG. 1C shows a dispatch table;
FIG. 1D is a schematic representation of apparatus for carrying out the Invention; and
FIG. 1E shows paths of execution through code.

FIG. 2A shows apparatus for carrying out the method of the invention
FIG. 2B shows a fragment of code including an exception; and
FIG. 2C shows a compiled fragment of code in accordance with the present invention.

FIG. 3A shows a section of code before compilation;
FIG. 3B shows a standard compilation of the code of FIG. 3A;
FIG. 3C shows compilation of code in accordance with a preferred embodiment;
FIG. 3D shows a code buffer;
FIG. 3E shows the memory arrangement in a computer system; and
FIG. 3F shows apparatus for carrying out the method of the invention.

FIG. 4A illustrates a hierarchical structure in object-oriented programming;
FIG. 4B shows the arrangement of data stored in dispatch tables;
FIG. 4C shows the application of an interface hash table to a dispatch table;
FIG. 4D is a hierarchical structure of a domestic equipment system;
FIG. 4E shows dispatch tables used in operating devices in the domestic system of FIG. 4D; and
FIG. 4F shows a controller program with driver devices for operating the devices in the domestic system of FIG. 4D.

FIG. 5A is a schematic illustration of data storage in a stack;
FIG. 5B shows an activation stack;
FIG. 5C illustrates how checks are made on references in a frame;
FIG. 5D shows the arrangement of data in a procedure call frame;
FIG. 5E shows the execution of a procedure; and
FIG. 5F shows the arrangement of the contents of a barrier descriptor block.

FIG. 6A illustrates the principle of a virtual machine;
FIG. 6B illustrates the operation of an emulator stack;
FIG. 6C illustrates the operation of a unified stack;
FIG. 6D shows an embodiment of the present invention; and
FIG. 6E shows an apparatus embodiment of the present invention.

FIG. 7A shows the division of memory according to a prior art approach;
FIG. 7B illustrates another prior art approach;
FIG. 7C shows an arrangement of objects in a so-called "concurrent" environment;
FIG. 7D shows the tracing of garbage collection work;
FIG. 7E shows the structure of an object;
FIG. 7F shows an empty stack;
FIG. 7G shows the structure of an individual packet according to the present invention; and
FIG. 7H shows the overall operation of the present invention.

FIG. 8A shows parts of a PC computer system for dealing with an interrupt;
FIG. 8B shows steps in the handling of an interrupt in an embodiment;
FIG. 8C illustrates code of an interrupt handler; and
FIG. 8D illustrates apparatus for carrying out an embodiment.

FIG. 9A shows a flow diagram illustrating a preferred embodiment;
FIG. 9B shows a section of compiled code;
FIG. 9C shows a different section of compiled code; and
FIG. 9D shows apparatus for carrying out a preferred embodiment.

FIG. 10A shows a link list;
FIG. 10B shows a looped link list;
FIG. 10C shows the movement of a pointer in a looped link list; and
FIG. 10D illustrates a preferred embodiment of apparatus; and
FIG. 10E shows a flow diagram in respect of a preferred embodiment in which a pointer is rewritten to point to an item in a list for facilitating access to the list.

FIG. 11A shows schematically the code buffer configuration of an embodiment; and
FIG. 11B shows schematically code fragments of an embodiment.

FIGS. 12A to 12D illustrate the use of patches in compiled code;
FIG. 12E is a flow diagram of a preferred method embodiment;
FIG. 12F illustrates the use of patches with potentially polymorphic methods; and
FIG. 12G is a block diagram of a preferred apparatus embodiment.